Graph Sparsification by Edge-Connectivity and Random Spanning Trees

Wai Shing Fung* Nicholas J. A. Harvey*

Abstract

We present new approaches to constructing graph sparsifiers — weighted subgraphs for which every cut has the same value as the original graph, up to a factor of $(1 \pm \epsilon)$. Our first approach independently samples each edge uv with probability inversely proportional to the edge-connectivity between u and v. The fact that this approach produces a sparsifier resolves a question posed by Benczúr and Karger (2002). Concurrent work of Hariharan and Panigrahi also resolves this question. Our second approach constructs a sparsifier by forming the union of several uniformly random spanning trees. Both of our approaches produce sparsifiers with $O(n \log^2(n)/\epsilon^2)$ edges. Our proofs are based on extensions of Karger's contraction algorithm, which may be of independent interest.

1 Introduction

Graph sparsification is an important technique in designing efficient graph algorithms. Different notions of graph sparsifiers have been considered in the literature. Roughly speaking, given a graph $G = (V, E)$, a sparsifier G' of G is a sparse subgraph of G that approximates G in some measures, e.g., pairwise distance, cut values, or the quadratic form defined by the graph Laplacian. G' may be weighted or not. Throughout this paper, we let $n = |V|$ and $m = |E|$. Since many graph algorithms have running times that depend on m, if G is dense, then the running time can be improved by replacing G with G', possibly with some loss in the quality of the solution.

Let us define a *cut sparsifier* to be a weighted subgraph that approximately preserves the value of every cut to within a multiplicative error of $(1 \pm \epsilon)$. The main motivation for cut sparsifiers was to improve the runtime of approximation algorithms for finding various sorts of cuts; indeed, they have been used extensively for this purpose [24, 4, 1, 27]. The first cut sparsifier was Karger's *graph skeleton* [23, 24]. He showed that sampling each edge independently with probability $p = \Theta(\log n/\epsilon^2 c)$, where c is the size of the min cut, gives a sparsifier of size $O(pm)$. Unfortunately, this is of little use when c is small. The celebrated work of Benczúr and Karger [4, 5] improved on this by using non-uniform sampling, obtaining a cut sparsifier with only $O(n \log n/\epsilon^2)$ edges. Their sparsifier is constructed by randomly sampling every edge with probability inversely proportional to its *edge strength*, and weighting the sampled edges accordingly.

Spielman and Teng [38, 39] define *spectral sparsifiers* — subgraphs that approximately preserve the quadratic form of the graph Laplacian. Such sparsifiers are stronger than the previously mentioned sparsifiers that only preserve cuts. Spielman and Teng's motivation for studying spectral sparsifiers was to use them as a building block for algorithms that solve linear systems in near-linear time [38, 36]. They construct spectral sparsifiers with $O(n \log^c n)$ edges, for some large constant

*University of Waterloo, Department of Combinatorics and Optimization. Supported by an NSERC Discovery Grant. Email: {wsfung,harvey}@uwaterloo.ca.

c. This was improved to $O(n \log n/\epsilon^2)$ edges by Spielman and Srivastava [37], by independently sampling edges according to their effective resistances.

Spielman and Srivastava conjectured that there exist spectral sparsifiers with $O(n/\epsilon^2)$ edges. Towards that conjecture, Goyal, Rademacher and Vempala [19] showed that sampling just two random spanning trees gives a cut sparsifier in bounded-degree graphs and random graphs. Finally, in a remarkable paper, Batson, Spielman and Srivastava [2] construct spectral sparsifiers with only $O(n/\epsilon^2)$ edges.

In this paper, we study several questions provoked by this previous work.

- Benczúr and Karger ask: Does sampling according to edge connectivity instead of edge strength give a sparsifier?
- The subgraph produced by Goyal, Rademacher and Vempala is an *unweighted* subgraph. If we sample random spanning trees and apply weights to the resulting edges, does this give a better sparsifier?
- Are there other approaches to achieving sparsifiers with $o(n \log n)$ edges?

In this paper, we give a positive answer to the first two questions. We also give a negative result on using random spanning trees to answer the third question. In concurrent, independent work, Hariharan and Panigrahi [20] also resolve the first question.

1.1 Notation

Before stating our results, we introduce some notation. For a multigraph $G = (V, E)$ with edge weights $u : E \to \mathbb{R}_+$ and a set $F \subseteq E$ of multiedges, the notation $u(F)$ denotes $\sum_{e \in F} u_e$. The notation $|F|$ denotes the total number of copies of all multiedges in F. For any set $S \subseteq V$, we define $\delta(S)$ to be the set of all copies of edges in E with exactly one end in S. So the notation $u(\delta(S))$ denotes the total weight of the cut $\delta(S)$.

For an edge $st \in E$, the (local) *edge connectivity* between s and t, denoted k_{st}, is defined to be the minimum weight of a cut that separates s and t. The *effective conductance* of edge st, denoted c_{st}, is the amount of current that flows when each edge e is viewed as a resistor of value u_e and a unit voltage difference is imposed between s and t. A *k-strong component* of G is a maximal k-edge-connected, vertex-induced subgraph of G. The *strength* of edge st, denoted by k'_{st}, is the maximum value of k such that a k-strong component of G contains both s and t.

Informally, all three of k_{st}, c_{st} and k'_{st} measure the connectivity between s and t. The values of k'_{st} and c_{st} are incomparable: k'_{st} can be $\Omega(n)$ times larger than c_{st} or vice versa. However $k_{st} \geq \max\{c_{st}, k'_{st}\}$ always holds. For more details, see Appendix A.

1.2 Our results

Theorem 1.1. *Let $G = (V, E)$ be a simple, weighted graph with edge weights $u : E \to \mathbb{Z}_+$. Let $\epsilon \geq 1/n$ and let $\rho := d \log^2 n$ where $d = O(1/\epsilon^2)$. For each edge e, let κ_e be a parameter such that $\kappa_e \leq k_e$. With high probability, the graph G' produced by Algorithm 1 satisfies*

$$|u(\delta(S)) - w(\delta(S))| \;\leq\; \epsilon \cdot u(\delta(S)) \qquad \forall S \subseteq V. \tag{1.1}$$

Furthermore, with high probability,

$$|E'| \;\leq\; 2\rho \sum_{e \in E} \frac{u_e}{k_e} + O(\log n). \tag{1.2}$$

Algorithm 1 A general algorithm for producing a sparsifier of G by sampling edges.

procedure Sparsify($G,\ u,\ \kappa$)
input: A graph $G = (V, E)$ with edge weights $u : E \to \mathbb{Z}_+$, and connectivity estimates κ
output: A graph $G' = (V, E')$ with edge weights $w : E \to \mathbb{R}_+$
For $i = 1, \ldots, \rho$
▷ *We refer to this as the i^{th} round of sampling*
For each $e \in E$
For $j = 1, \ldots, u_e$
With probability $p_e = 1/\kappa_e$, add edge e to E' (if it does not already exist) and increase w_e by κ_e/ρ
Return G' and w

This theorem is proven in Sections 2 and 3. The condition that $\epsilon \geq 1/n$ is not really restrictive because if $\epsilon < 1/n$ then the theorem is trivial: G is itself a sparsifier with $O(1/\epsilon^2)$ edges. The condition that the edge weights are integral is not restrictive either. If the edge weights are any positive real numbers then they can be approximated arbitrarily well by rational numbers, and these rationals can be scaled up to integers. This does not affect the conclusion of Theorem 1.1, as it does not depend on the magnitude of u.

The sampled weight of each copy of edge e is a binary random variable that takes value κ_e/ρ with probability $1/\kappa_e$ and zero otherwise. When $\kappa_e = k_e$, this random variable has the highest variance, and therefore the cuts of G' are least concentrated. So, at least intuitively, the theorem is hardest to prove when $\kappa_e = k_e$, and the result for smallest κ_e values will follow as a corollary. This intuition is indeed correct, and we obtain several interesting corollaries of Theorem 1.1 by invoking Algorithm 1 with different κ_e values. Proofs are in Appendix B.

Corollary 1.2. *Let $\kappa_e = k_e$. Then* (1.1) *holds and $|E'| = O(n \log^2 n/\epsilon^2)$ with high probability.*

Corollary 1.3. *Let $\kappa_e = c_e$. Then* (1.1) *holds and $|E'| = O(n \log^2 n/\epsilon^2)$ with high probability.*

Spielman and Srivastava [37] prove a related result: taking $\kappa_e = c_e$, only $O(\log n/\epsilon^2)$ rounds of sampling suffice for (1.1) to hold with *constant* probability. A simple modification of their proof implies Corollary 1.3. (See, e.g., Koutis et al. [30].) It is unclear whether $O(\log n/\epsilon^2)$ rounds suffice for (1.1) to hold with high probability.

Corollary 1.4. *Let $\kappa_e = k'_e$. Then* (1.1) *holds and $|E'| = O(n \log^2 n/\epsilon^2)$ with high probability.*

Benczúr and Karger [5] prove a stronger result: taking $\kappa_e = k'_e$, only $O(\log n/\epsilon^2)$ rounds of sampling suffice for (1.1) to hold with high probability.

An important aspect of our analysis is that we rely only on Chernoff bounds. In contrast, Spielman and Srivastava [37] use sophisticated concentration bounds for matrix-valued random variables. An advantage of Chernoff bounds is that they are very flexible and have been generalized in many ways. This flexibility enables us to prove the following result in Section 4.

Theorem 1.5. *Let $G = (V, E, u)$ be a graph with edge weights $u : E \to \mathbb{Z}_+$, let $\epsilon \geq 1/n$, and let $\rho := d \log^2 n$ where $d = O(1/\epsilon^2)$. With high probability, the graph G' produced by Algorithm 2 satisfies*

$$|u(\delta(S)) - w(\delta(S))| \ \leq \ \epsilon \cdot u(\delta(S)) \qquad \forall S \subseteq V.$$

Clearly $|E'| \leq \rho(n-1) = O(n \log^2 n/\epsilon^2)$.

Algorithm 2 An algorithm for producing a sparsifier of G by sampling random spanning trees.

procedure SparsifyByTrees($G,\ u$)
input: A graph $G = (V, E)$ with edge weights $u : E \to \mathbb{Z}_+$
output: A graph $G' = (V, E')$ with edge weights $w : E \to \mathbb{R}_+$
For each $e \in E$, compute the conductance c_e
For $i = 1, \ldots, \rho$
$\triangleright$ *We refer to this as the i^{th} round of sampling*
Let T be a uniformly random spanning tree
For each $e \in T$
Add edge e to E' (if it does not already exist) and increase w_e by c_e/ρ
Return G' and w

Counting Small Cuts. An important ingredient in the proof of Theorem 1.1 is an extention of Karger's random contraction algorithm for computing global minimum cuts [22, 25]. We describe two variants of this algorithm which introduce the additional ideas of splitting off vertices and performing random walks. The main purpose of these variants is to prove generalizations of Karger's cut-counting theorem [22, 25], which states that the number of cuts of size at most α times the minimum is less than $n^{2\alpha}$. Our generalizations give "Steiner variants" of this theorem. Roughly speaking, we show that, amongst all cuts that separate a certain set of terminals, the number of size at most α times the minimum is less than $n^{2\alpha}$.

Since our cut-counting result may be of independent interest, we state it formally now.

Theorem 1.6. *Let $G = (V, E)$ be a graph and let $B \subseteq E$ be arbitrary. Suppose that $k_e \geq K$ for every $e \in B$. Then, for every real $\alpha \geq 1$,*

$$|\{\ \delta(S) \cap B\ :\ S \subseteq V\ \wedge\ |\delta(S)| \leq \alpha K\ \}|\ <\ n^{2\alpha}.$$

We discuss this theorem in further detail in Section 3. For now, let us only mention that this theorem reduces to Karger's cut-counting theorem by setting $B = E$ and setting K to the global minimum cut value. In this special case, it states that the number of α-minimum cuts is at most $n^{2\alpha}$. In concurrent, independent work, Hariharan and Panigrahi [20] have also proven Theorem 1.6.

1.3 Algorithms

In this section we describe several algorithms to efficiently construct sparsifiers. To make Algorithm 1 into a complete algorithm, the most challenging step is to efficiently compute the κ_e values. This can be done by computing estimates for either k_e, c_e, or k'_e.

Edge Connectivity. The simplest approach is to estimate k_e. Several methods for computing such estimates can be found in the work of Benczúr and Karger [5]. In fact, these methods can be significantly simplified because they were originally designed for estimating k'_e, which is more challenging to estimate than k_e. The following theorems describe how we can combine these methods with Algorithm 1 to efficiently construct sparsifiers. The resulting algorithms are simple enough for a real-world implementation, and they have theoretical value too: they can be used to improve the $O(m \log^3 n)$ running time of Benczúr and Karger's sampling algorithm to nearly linear time (cf. Theorem 1.9).

These algorithmic results were also described in the earlier work Hariharan and Panigrahi [20]. In fact, their runtime bounds are slightly better, due to a different method of analysis.

Theorem 1.7. *Given a graph G and edge weights $u : E \to \mathbb{Z}_+$ where $\max_e u_e = \text{poly}(n)$, a sparsifier G' of size $O(n \log^3 n/\epsilon^2)$ that satisfies (1.1) can be computed in $O(m + n \log^5 n/\epsilon^4)$ time. If G is simple, the time complexity can be reduced to $O(m)$.*

Theorem 1.8. *Given a graph G and edge weights $u : E \to \mathbb{Z}_+$ where $\max_e u_e = \text{poly}(n)$, a sparsifier G' of size $O(n \log^2 n/\epsilon^2)$ that satisfies (1.1) can be computed in $O(m \log^2 n + n \log^4 n/\epsilon^4)$ time.*

Combining Theorems 1.7 and 1.8 with Benczúr and Karger's algorithm, we can obtain the following result.

Theorem 1.9. *Given a graph G and edge weights $u : E \to \mathbb{Z}_+$ where $\max_e u_e = \text{poly}(n)$, a sparsifier G' of size $O(n \log n/\epsilon^2)$ that satisfies (1.1) can be computed in $O(m + n \log^5 n/\epsilon^4)$ time.*

Proofs of these theorems can be found in Appendix H.

Effective Conductance. As described above, Spielman and Srivastava [37] also construct sparsifiers by sampling according to the effective conductances. Moreover, they describe an algorithm to approximate the effective conductances in $m \log^{O(1)} n$ time. This algorithm can be implemented more efficiently using the recent simplified method of Koutis, Miller and Peng [30]. Combining this with Algorithm 1, we can construct a sparsifier with $O(n \log^2 n/\epsilon^2)$ edges in $\tilde{O}(m \log^3 n)$ time.

Random Spanning Trees. Algorithm 2 can also be implemented efficiently, although we do not know how to do this in nearly linear time. The best known algorithms for sampling (approximately) uniform spanning trees run in $\tilde{O}(\min\{m\sqrt{n}, n^{2.376}\})$ time [8, 26]. Combining this with the method described above for approximating the effective conductances gives an algorithm to compute sparsifiers with $O(n \log^2 n/\epsilon^2)$ edges. The running time of this algorithm is dominated by the time needed to sample $O(\log^2 n/\epsilon^2)$ random spanning trees. Although this algorithm is not as efficient as those listed above, the numerous special properties of random spanning trees might make it useful in other ways.

1.4 Limits of sparsification

In Corollaries 1.2, 1.3 and 1.4, the number of rounds of sampling ρ cannot be decreased to $o(\log n)$. To see this, consider a path of length n — with probability tending to 1 the sampled graph would be disconnected and hence not approximate the original graph.

Sampling random spanning trees overcomes this obstacle since the graph is connected with probability 1. Indeed, Goyal, Rademacher and Vempala [19] show that, for any constant-degree graph, the unweighted union of just 2 spanning trees approximates every cut to within a factor $O(\log n)$. In Section 4.1 we prove the following negative result for sampling random spanning trees.

Lemma 1.10. *For any constant $c \geq 1$, there is a graph such that Algorithm 2 requires $\rho = \Omega(\log n)$ to approximate all cuts within a factor c with constant probability.*

2 Sparsifiers by independent sampling

In this section we prove our main result, Theorem 1.1. Perhaps the most natural approach would be to analyze the probability of poorly sampling each cut, then union bound over all cuts. In Appendix C we explain why this simple approach fails, why Benczúr and Karger [5] proposed to decompose the graph and separately analyze the pieces, and why their approach does not suffice to prove Theorem 1.1.

Our analysis also involves partitioning the graph, but using a different approach. In fact, a very similar partitioning was used in an earlier proof of Benczúr and Karger [4, §3.2] [3, §9.3.2]. We will partition the graph into subgraphs, each consisting of edges with roughly equal values of k_e. Formally, we partition G into subgraphs with edge sets $E_1, E_2, \ldots$, where

$$E_i \;=\; \left\{\, e \in E \,:\, 2^i \le k_e < 2^{i+1} \,\right\}.$$

We emphasize that E_i is defined using k_e, not κ_e.

To prove that the weights of all cuts are nearly preserved (i.e., that Eq. (1.1) holds), we will use a Chernoff bound to analyze the error contributed to each cut by each subgraph E_i. A union bound allows us to analyze the probability of large deviation for all cuts simultaneously. As in previous work [24, 5], the key to making this union bound succeed is to show that most cuts are very large, so their probability of deviation is very small. This is achieved by our cut-counting theorem, Theorem 1.6.

From this point onwards, to simplify our notation, we will no longer think of G as a weighted graph, but rather think of it as an unweighted multigraph which has u_e parallel copies of each edge e. The main benefit of this change is that the total weight of a cut can now be written $|\delta(S)|$ instead of $u(\delta(S))$ since, for multigraphs, the notation $|\delta(S)|$ gives the total number of copies of all multiedges in $\delta(S)$. We hope that this choice of notation makes the following proofs easier to read.

The crucial definition for this paper is as follows. We say that a non-empty set of edges $F \subseteq E_i$ is *induced by a cut* C if $F = C \cap E_i$. Any such set F is called a *cut-induced* subset of E_i. Note that F could be induced by different cuts C and C', i.e., $C \cap E_i = F = C' \cap E_i$. For a cut-induced set $F \subseteq E_i$, define

$$q(F) \;:=\; \min\left\{\, |\delta(S)| \,:\, S \subseteq V \;\wedge\; \delta(S) \cap E_i = F \,\right\}. \tag{2.1}$$

So $q(F)$ is the minimum size[1] of a cut that induces F. This is an important definition since the amount of error we can allow when sampling F is naturally constrained by the smallest cut which induces F.

We also define a "normalized" form of $q(F)$, which is $\alpha(F) := q(F)/2^i$. Note that 2^i is a lower bound on the size of any cut that intersects F, because every edge $e \in E_i$ has $2^i \le k_e$. So we can think of $\alpha(F)$ as a quantity that measures how close $q(F)$ is to the minimum size of any cut that intersects E_i. Clearly $\alpha(F) \ge 1$.

For any set F of edges, the random variable X_F denotes the total weight of all sampled copies of the edges in F, over all rounds of sampling. The main challenge in proving Theorem 1.1 is to prove concentration for all X_F where F is a cut-induced subset of some E_i.

[1] We remind the reader that the notation $|\delta(S)|$ implicitly involves the edge multiplicities. So, thinking of G as a weighted graph, $q(F)$ is really the minimum weight of a cut that induces F.

2.1 The bad events

Let $F \subseteq E_i$ be a cut-induced set. We now define three bad events which indicate that the edges in F were not sampled well. The first two events are:

$$\mathcal{A}_F : \quad |X_F - |F|| > \epsilon |F|$$
$$\mathcal{B}_F : \quad |X_F - |F|| > \frac{\epsilon q(F)}{\ln n}$$

The third event is not needed to analyze unweighted graphs (i.e., if every edge multiplicity is $u_e = 1$); it is only needed to deal with arbitrary weights. The third event is

$$\mathcal{C}_F : \quad X_F - |F| \; > \; g^{-1}\left(\frac{\epsilon^2 q(F)}{|F| \ln n}\right) \cdot |F|,$$

where $g : \mathbb{R} \to \mathbb{R}$ is the function defined by

$$g(x) \; = \; (1+x)\ln(1+x) - x. \tag{2.2}$$

Note that its derivative is $g'(x) = \ln(1+x)$, so g is strictly monotonically increasing on $\mathbb{R}_+$, and hence invertible. Furthermore, g^{-1} is also strictly monotonically increasing on $\mathbb{R}_+$, by the inverse function theorem of calculus.

We will show that, assuming that these events do not hold (for certain cut-induced sets), then the weights of all cuts are approximately preserved. To this end, we bound the probability of the bad events by the following three claims. These claims are proven by straightforward applications of Chernoff bounds in Appendix D. Recall that the parameter d in the statement of the claims satisfies $d = O(1/\epsilon^2)$, as stated in Theorem 1.1.

Claim 2.1. *Let $F \subseteq E_i$ be a cut-induced set with $q(F) \leq |F| \ln n$. Then*

$$\Pr[\,\mathcal{A}_F\,] \; \leq \; 2n^{-d\alpha(F)\epsilon^2/6}.$$

Claim 2.2. *Let $F \subseteq E_i$ be a cut-induced set with $q(F) > |F| \ln n$. Then*

$$\Pr[\,\mathcal{B}_F\,] \; \leq \; 2n^{-d\alpha(F)\epsilon^2/8}.$$

Claim 2.3. *For every cut-induced set F,*

$$\Pr[\,\mathcal{C}_F\,] \leq n^{-d\alpha(F)\epsilon/2}.$$

Claim 2.4. *By choosing $d = O(1/\epsilon^2)$ sufficiently large, then with high probability, every cut-induced set F satisfies*

- *if $q(F) \leq |F| \ln n$ then $\mathcal{A}_F$ does not hold;*
- *if $q(F) > |F| \ln n$ then $\mathcal{B}_F$ does not hold; and*
- *$\mathcal{C}_F$ does not hold.*

The proof of Claim 2.4, given in Appendix D, is a straightforward modification of an argument of Karger [24]. The only difference with our proof is that we require a result which bounds the number of small cut-induced sets. Such a statement is given by Corollary 2.5, which follows directly from Theorem 1.6.

Corollary 2.5. *For each i and any real number $\alpha \geq 1$, the number of cut-induced sets $F \subseteq E_i$ with $\alpha(F) \leq \alpha$ is less than $n^{2\alpha}$.*

Proof. Since every $e \in E_i$ satisfies $k_e \geq 2^i$, we may apply Theorem 1.6 with $B = E_i$ and $K = 2^i$. This yields

$$|\{\ \delta(S) \cap E_i\ :\ S \subseteq V\ \wedge\ |\delta(S)| \leq \alpha 2^i\ \}|\ <\ n^{2\alpha}.$$

Now, by the definition of $\alpha(F)$ and $q(F)$, for every cut-induced set $F \subseteq E_i$ with $\alpha(F) \leq \alpha$, there exists a cut $\delta(S)$ such that $\delta(S) \cap E_i = F$ and $|\delta(S)| \leq \alpha 2^i$. This proves the desired statement. ■

2.2 All cuts are preserved

In this section we prove that Eq. (1.1) holds. Recall that the random variable X_C denotes total weight of all sampled edges in C. Our main lemma is

Lemma 2.6. *With high probability, every cut C satisfies $|X_C - |C|| = O(\epsilon|C|)$.*

For unweighted graphs, the proof of this lemma is quite simple. For weighted graphs, we require the following three technical claims, which are proven in Appendix E.

Claim 2.7. *Let $F \subseteq E_i$ be a cut-induced set. Then $|F| < n^2 2^i$.*

Claim 2.8. *For any integer $d \geq 0$,*

$$\sum_{i \leq \lg|C| - 2\lg n - d} |C \cap E_i|\ <\ 2^{1-d}\,|C|.$$

Claim 2.9. *Define $h : \mathbb{R} \to \mathbb{R}$ by*

$$h(x)\ =\ \frac{2x}{\ln(1+\sqrt{x})}.$$

Then $g^{-1}(x) \leq h(x)$ for all $x \geq 0$.

Proof (of Lemma 2.6). We wish to prove that $|X_C - |C|| \leq O(\epsilon|C|)$. We may assume that the conclusions of Claim 2.4 hold, since they hold with high probability. We use those facts to bound the error contribution from each cut-induced set $C_i := C \cap E_i$.

To perform the analysis, we partition the C_i's into three classes, according to which bad event ($\mathcal{A}_{C_i}$, $\mathcal{B}_{C_i}$ or $\mathcal{C}_{C_i}$) will be used to analyze the error. This partitioning depends on the threshold

$$t\ :=\ \lg|C| - 4\lg n - \lg(1/\epsilon).$$

The sets of indices are

$$\begin{aligned}
\mathcal{I}_1 &= \{\ i\ :\ |C_i| \geq q(C_i)/\ln n\ \} \\
\mathcal{I}_2 &= \{\ i\ :\ 0 < |C_i| < q(C_i)/\ln n\ \wedge\ t < i \leq \lg|C|\ \} \\
\mathcal{I}_3 &= \{\ i\ :\ 0 < |C_i| < q(C_i)/\ln n\ \wedge\ 0 \leq i \leq t\ \}.
\end{aligned}$$

We remark that

$$\lg|C| - t\ \leq\ 5\lg n \tag{2.3}$$

since we assume that $\epsilon \geq 1/n$.

To analyze the error $X_C - |C|$, we expand it as a sum over cut-induced sets.

$$X_C - |C|\ =\ \sum_{i\in\mathcal{I}_1}(X_{C_i} - |C_i|)\ +\ \sum_{i\in\mathcal{I}_2}(X_{C_i} - |C_i|)\ +\ \sum_{i\in\mathcal{I}_3}(X_{C_i} - |C_i|) \tag{2.4}$$

Unweighted graphs. For unweighted graphs, the analysis is simple. Since any cut satisfies $|C| < n^2$, we have $\lg |C| \leq 2 \lg n$ and so $\mathcal{I}_3 = \emptyset$. We have assumed that the conclusions of Claim 2.4 hold, so the events $\mathcal{A}_{C_i}$ and $\mathcal{B}_{C_i}$ do not occur (under the stated conditions on $q(C_i)$). Therefore

$$\begin{aligned}
|X_C - |C|| &\leq \sum_{i \in \mathcal{I}_1} |X_{C_i} - |C_i|| + \sum_{i \in \mathcal{I}_2} |X_{C_i} - |C_i|| \\
&\leq \sum_{i \in \mathcal{I}_1} \epsilon |C_i| + \sum_{i \in \mathcal{I}_2} \frac{\epsilon q(C_i)}{\ln n} \\
&\leq \epsilon |C| + \frac{\epsilon |C| |\mathcal{I}_2|}{\ln n} \qquad \text{(since } q(C_i) \leq |C|\text{)} \\
&= O(\epsilon |C|) \qquad \text{(by Eq. (2.3)).}
\end{aligned} \tag{2.5}$$

This completes the proof for the unweighted case.

Weighted graphs. For weighted graphs the analysis is slightly more complicated because the number of cut-induced sets C_i that contribute error may be much larger than $\lg n$, because $\mathcal{I}_3$ may be non-empty. To show that the total error is still small, we will need to use the events $\mathcal{C}_{C_i}$.

Consider Eq. (2.4) again. The first two sums were analyzed in Eq. (2.5), so it suffices to analyze the third sum. First we prove a lower bound on this sum:

$$\sum_{i \in \mathcal{I}_3} (|C_i| - X_{C_i}) \leq \sum_{i \in \mathcal{I}_3} |C_i| = O(\epsilon |C|),$$

by Claim 2.8 and the definition of t.

Now we prove an upper bound. By Claim 2.4, we may assume that the events $\mathcal{C}_{C_i}$ do not hold.

$$\sum_{i \in \mathcal{I}_3} (X_{C_i} - |C_i|) \leq \sum_{i \in \mathcal{I}_3} g^{-1}\left(\frac{\epsilon^2 q(C_i)}{|C_i| \ln n}\right) \cdot |C_i|$$

We use the fact that g^{-1} is monotonically increasing and that $|C| \geq q(C_i)$. This yields

$$\begin{aligned}
&\leq \sum_{i \in \mathcal{I}_3} g^{-1}\left(\frac{\epsilon |C|}{|C_i| \ln n}\right) \cdot |C_i| \\
&\leq \sum_{i \in \mathcal{I}_3} h\left(\frac{\epsilon |C|}{|C_i| \ln n}\right) |C_i| \qquad \text{(by Claim 2.9)} \\
&= O\left(\frac{\epsilon |C|}{\ln n}\right) \sum_{i \in \mathcal{I}_3} \frac{1}{\log\left(1 + \sqrt{\frac{\epsilon |C|}{|C_i| \ln n}}\right)} \\
&\leq O\left(\frac{\epsilon |C|}{\log n}\right) \sum_{i \in \mathcal{I}_3} \frac{1}{\lg\left(\frac{\epsilon |C|}{n^2 \lg(n) 2^i}\right)} \qquad \text{(by Claim 2.7)} \\
&= O\left(\frac{\epsilon |C|}{\log n}\right) \sum_{t-i \in \mathcal{I}_3} \frac{1}{\lg\left(\frac{\epsilon |C|}{n^2 \lg(n) 2^{t-i}}\right)} \\
&\leq O\left(\frac{\epsilon |C|}{\log n}\right) \sum_{t-i \in \mathcal{I}_3} \frac{1}{i + \lg\left(\frac{\epsilon |C|}{n^2 \lg(n) 2^t}\right)}
\end{aligned}$$

$$\leq O\left(\frac{\epsilon|C|}{\log n}\right) \sum_{t-i\in\mathcal{I}_3} \frac{1}{i+\lg n}. \tag{2.6}$$

The last inequality holds since

$$\frac{\epsilon|C|}{n^2\lg(n)2^t} = \frac{\epsilon|C|}{n^2\lg(n)2^{\lg|C|-4\lg n-\lg(1/\epsilon)}} \geq \frac{n^2}{\lg n}.$$

The sum in Eq. (2.6) is a subseries of a harmonic series with at most n^2 terms (since there are at most $\binom{n}{2}$ distinct κ_e values) so the value of this sum is $O(\log n)$. Thus we have shown that the third sum in Eq. (2.4) is at most $O(\epsilon|C|)$. ■

2.3 The size of the sparsifier

To complete the proof of Theorem 1.1, it remains to show that G' does not have too many edges, i.e., Eq. (1.2) holds. We have

$$\begin{aligned}
\mathrm{E}\left[\,|E'|\,\right] &= \sum_{\text{edge } e\in E} \Pr\left[\,\text{at least one copy of } e \text{ is sampled in one of the rounds}\,\right] \\
&= \sum_{\text{edge } e\in E} 1-\left(1-\frac{1}{\kappa_e}\right)^{\rho u_e} \\
&\leq \rho \sum_{\text{edge } e\in E} \frac{u_e}{\kappa_e},
\end{aligned}$$

so the right-hand side of Eq. (1.2) is at least $2\,\mathrm{E}\left[\,|E'|\,\right]+O(\log n)$. By a simple Chernoff bound, it follows that Eq. (1.2) holds with high probability.

3 The cut-counting theorem

In this section, we prove Theorem 1.6, which is our generalization of Karger's cut-counting theorem [22, 25]. The proof of Karger's theorem is based on his randomized contraction algorithm for finding a global minimum cut of a graph. Roughly speaking he shows that, for any small cut-induced set, it has non-negligible probability of being output by the algorithm, and hence the number of small cut-induced sets must be small. We will prove our generalized cut-counting theorem by analyzing a variant of the contraction algorithm which incorporates the additional idea of splitting off vertices.

The formal statement of our theorem is:

Theorem 1.6. *Let $G=(V,E)$ be a graph and let $B\subseteq E$ be arbitrary. Suppose that $k_e\geq K$ for every $e\in B$. Then, for every real $\alpha\geq 1$,*

$$|\{\ \delta(S)\cap B\ :\ S\subseteq V\ \wedge\ |\delta(S)|\leq \alpha K\ \}|\ <\ n^{2\alpha}.$$

This theorem becomes easier to understand by restating it using the terminology of Section 2 (cf. Corollary 2.5). A cut-induced subset of B is precisely a set of the form $\delta(S)\cap B$, so the theorem is counting cut-induced sets satisfying some condition. This condition is: for a cut-induced set F, there must exist $S\subseteq V$ with $|\delta(S)|\leq\alpha K$ and $F=\delta(S)\cap B$. This condition is equivalent to $q(F)\leq\alpha K$, where q is the function defined in Eq. (2.1). So the conclusion of Theorem 1.6 can be restated as

$$|\{\ F\ :\ F \text{ is a cut-induced subset of } B\ \wedge\ q(F)\leq\alpha K\ \}|\ <\ n^{2\alpha}.$$

Comparison to Karger's theorem. For the sake of comparison, Karger's theorem is:

Theorem 3.1 (Karger [22, 25]). *Let $G = (V, E)$ be a connected graph and let $B \subseteq E$ be arbitrary. Suppose that the value of the global minimum cut is at least K. Then, for every real $\alpha \geq 1$,*

$$|\{\ C\ :\ C \text{ is a cut in } G\ \wedge\ |C| \leq \alpha K\ \}|\ <\ n^{2\alpha}.$$

Our theorem improves on Karger's theorem in two ways. First of all, we count cut-induced sets instead of cuts. This is clearly more general and, as we mentioned before, it is useful because it avoids overcounting cut-induced sets that are shared by many cuts. Secondly, we want to bound the number of "small" cut-induced sets in B. The bounds given by both theorem are $n^{2\alpha}$, where α measures how small a cut or a cut-induced set is. However in our cut-counting lemma, α is relative to K, the size of a smallest cut that intersects with B, not relative to the size of a global minimum cut as in Karger's theorem. This is an improvement since the global minimum cuts may not intersect B at all, so the global minimum cut value could be much smaller than K.

For concreteness, consider the example in Figure 1, which appears in Appendix A. Suppose we want to bound the number of cuts that intersect with $E_{\lg n}$. (Here $E_{\lg n}$ consists of the single edge st.) Note that all such cuts have size $n-1$. However the global minimum cuts all have size 2, and they do not intersect with $E_{\lg n}$. From Theorem 3.1 we see that there are at most n^{n-1} cuts of size at most $n-1$ that intersect $E_{\lg n}$. In contrast, Theorem 1.6 states that there are at most n^2 cut-induced subsets of $E_{\lg n}$ that are induced by cuts of size at most $n-1$.

A weaker theorem based on effective conductance. We have also proven a weaker version of Theorem 1.6 which does not suffice to prove Theorem 1.1 but does suffice to prove Corollary 1.3. This weaker version is stated as Theorem 3.2; it is weaker because the hypothesis $c_e \geq K$ is stronger than the hypothesis $k_e \geq K$, by Claim A.1.

Theorem 3.2. *Let $G = (V, E)$ be a graph and let $B \subseteq E$ be arbitrary. Suppose that $c_e \geq K$ for every $e \in B$. Then, for every real $\alpha \geq 1$,*

$$|\{\ F\ :\ F \text{ is a cut-induced subset of } B\ \wedge\ q(F) \leq \alpha K\ \}|\ <\ n^{2\alpha}.$$

Although this theorem is weaker than Theorem 1.6, we feel that it is worth including in this paper because its proof is based on analysis of random walks that may be of independent interest. The proof is in Appendix G.

3.1 The generalized contraction algorithm

Theorem 1.6 follows immediately from Theorem 3.3, which is the analysis of our generalized contraction algorithm (Algorithm 3). Henceforth, we will use the following terminology. The edges in B are called *black*. Also, a cut is *black* if it contains a black edge, and a vertex is black if it is incident to a black edge. An edge, vertex or cut is *white* if it is not black.

Theorem 3.3. *For any cut-induced set $F \subseteq B$ with $q(F) \leq \alpha K$, Algorithm 3 outputs F with probability at least $n^{-2\alpha}$.*

Algorithm 3 is essentially the same as Karger's contraction algorithm, except that it maintains the invariant that G has no white vertex by *splitting-off*. For a pair of edges uv and vw, splitting-off uv and vw is the operation that removes uv and vw then adds a new edge uw. This splitting-off operation is *admissible* if it does not decrease the (local) edge connectivity between any pair of vertices s and t, except of course when one of those vertices is v. Splitting-off has many applications in solving connectivity problems because of the following theorem.

Algorithm 3 An algorithm for finding a small cut-induced set by splitting off white vertices.

procedure Contract($G,\ B,\ \alpha$)

input: A graph $G = (V, E)$, a set $B \subseteq E$, and an approximation factor α

output: A cut-induced subset of B

While there are more than $\lceil 2\alpha \rceil$ vertices remaining

  While there exists a white vertex v

    Perform admissible splitting-off at v until v becomes an isolated vertex

    Remove v

  Pick an edge e uniformly at random

  Contract e and remove any self loops

Pick a non-empty proper subset S of V uniformly at random and output the black edges with exactly one endpoint in S

Theorem 3.4 (Mader [33]). *Let $G = (V, E)$ be a connected graph and $v \in V$ be a vertex. If v has degree $\neq 3$ and v is not incident to any cut edge, then there is a pair of edges uv and vw such that the splitting-off of uv and vw is admissible.*

Since Algorithm 3 needs to perform admissible splitting-off, we must ensure that the hypotheses of Mader's theorem are satisfied. This can be accomplished by the simple trick of duplicating every edge, which ensures that G is Eulerian and its components are 2-edge-connected. Note that these conditions are preserved under all modifications to the graph performed by Algorithm 3, namely contraction, splitting-off and removal of self loops.

To prove Theorem 3.3, we fix a cut-induced set F with $q(F) \leq \alpha K$. We will show that, with good probability, the algorithm maintains the following invariants.

(I1): F is a cut-induced set in the remaining graph,

(I2): $q(F) \leq \alpha K$ (where $q(F)$ now minimizes over cuts in the remaining graph), and

(I3): every remaining black edge e satisfies $k_e \geq K$.

The only modifications to the graph made by Algorithm 3 are splitting-off operations, contraction of edges, and removal of self-loops. Clearly removing self-loops does not affect the invariants.

Claim 3.5. *The admissible splitting-off operations at v do not affect the invariants.*

Proof. For (I1), note that splitting-off affects only white edges, and all edges in F are black. For (I2), note that splitting-off only decreases the size of any cut. For (I3), the edge connectivity between any two black vertices is unaffected since the splitting-off is admissible and v is white. ■

Claim 3.6. *Let the number of remaining vertices be r. Assuming that the invariants hold, they will continue to hold after the contraction operation with probability at least $1 - 2\alpha/r$.*

Proof. For (I3), note that any black cut which exists after the contraction also existed before the contraction, so the edge connectivity between any two black vertices cannot decrease.

Now, with respect the graph before the contraction, let C be a minimum cardinality cut that induces F, i.e., $|C| = q(F)$. We claim that $\Pr[e \in C] \geq 2\alpha/r$, where the probability is over the random choice of e to be contracted. To see this, note that every remaining vertex u is black, so the cut $\delta(\{u\})$ is a black cut. By invariant (I3) we have $\delta(\{u\}) \geq K$, so the number of remaining

edges is at least $Kr/2$. Since e is picked uniformly at random,

$$\Pr[e \in C] \;\leq\; \frac{|C|}{Kr/2} \;=\; \frac{2q(F)}{Kr} \;\leq\; \frac{2\alpha}{r},$$

by (I2). Let us assume that $e \notin C$. Then F is still induced by C after contracting e, so (I1) is preserved. Furthermore, (I2) is preserved since $|C| \leq \alpha K$. ■

The following claim completes the proof of Theorem 3.3. We relegate its proof to Appendix F as it is the same argument used to prove Karger's theorem [22, 25]. (See also Karger [24, App. A], where a slightly more general result is proven.)

Claim 3.7. *The probability that Algorithm 3 outputs F is at least $n^{-2\alpha}$.*

3.2 Remarks on cactus representations

A special case of Theorem 3.1 is that any connected graph $G = (V, E)$ has at most n^2 (non-trivial) minimum cuts. (In fact, the theorem actually proves a bound of $\binom{n}{2}$, which is tight.) The same fact is implied by a much earlier result of Dinic, Karzanov and Lomonosov [10], which states that the minimum cuts have a *cactus representation*. Fleiner and Frank [15] give a recent exposition of this result.

Dinitz[2] and Vainshtein [11, 12] generalized this result as follows. (See also Fleiner and Jordán [16].) Let $U \subseteq V$ be a subset of vertices with $|U| \geq 2$. A cut $C = \delta(S)$ is called a *U-cut* if the partition $\{U \cap S, U \setminus S\}$ of U that it induces has both parts non-empty. A U-cut C is called minimal if $|C|$ is minimal amongst all U-cuts. Two minimal U-cuts are called equivalent if they induce the same partition of U. Dinitz and Vainshtein showed that the equivalence classes of minimal U-cuts have a cactus representation. In particular, there are at most n^2 equivalence classes of minimal U-cuts.

We now explain how the latter result also follows from Theorem 1.6. Let K be the minimum cardinality of a U-cut. We add dummy edges of weight ϵ/n^2 between all pairs of U-vertices and let B be the set of dummy edges. Then every dummy edge e has $k_e \geq K$ and every minimal U-cut has weight at most $(1+\epsilon)K$. By Theorem 1.6, the number of cut-induced sets induced by cuts of size at most $(1+\epsilon)K$ is at most $n^{2(1+\epsilon)}$. Any two equivalent minimal U-cuts induce the same cut-induced subset of B, so the number of equivalence classes is at most $n^{2(1+\epsilon)}$. Taking $\epsilon \to 0$ proves that there are at most n^2 equivalence classes of minimal U-cuts.

4 Sparsifiers by uniform random spanning trees

In this section we describe an alternative approach to constructing a graph sparsifier. Instead of sampling edges independently at random, as was done in Section 2, we will sample edges by picking random spanning trees. The analysis of this sampling proves Theorem 1.5. The proof is a small modification of the proof in Section 2, with some differences to handle the dependence in the sampled edges. The following two lemmas explain why sampling random spanning trees is similar to sampling according to effective conductances.

Let us introduce some notation. For an edge $st \in E$, we denote by r_{st} the effective resistance between s and t. This is the inverse of the effective conductance c_{st}.

[2] E. Dinic and Y. Dinitz are two different transliterations of the same person's name.

Lemma 4.1. *Let $G = (V, E)$ be an unweighted simple graph, and let T be a spanning tree in G chosen uniformly at random. Let $e_1, \ldots, e_k \in E$ be distinct edges. Then*

$$\Pr[e_1, \ldots, e_k \in T] \leq \Pr[e_1 \in T] \cdots \Pr[e_k \in T] \tag{4.1}$$

$$\Pr[e_1, \ldots, e_k \notin T] \leq \Pr[e_1 \notin T] \cdots \Pr[e_k \notin T] \tag{4.2}$$

Proof. In the case $k = 2$, Equation 4.1 was known to Brooks et al. [7, Equation (2.34)]. See also Lyons and Peres [32, Exercise 4.3]. For general k, this is a consequence of Theorem 4.5 in Lyons and Peres [32], which is a result of Feder and Mihail [14]. See also Goyal, Rademacher and Vempala [19, Section 3]. ■

One useful consequence of Lemma 4.1 is that concentration inequalities can be proven for the number of edges in T that lie in any given subset. The concentration is due to the following theorem:

Theorem 4.2. *Let $a_1, a_2, \ldots, a_k$ be reals in $[0, 1]$, and let $X_1, \ldots X_k$ be $\{0, 1\}$-valued random variables. Suppose that*

$$\Pr[\wedge_{i \in I} X_i = 1] \leq \prod_{i \in I} \Pr[X_i = 1] \qquad \forall I \subseteq [k]$$

$$\Pr[\wedge_{i \in I} X_i = 0] \leq \prod_{i \in I} \Pr[X_i = 0] \qquad \forall I \subseteq [k].$$

Suppose $\mu_1 \leq E[\sum_i a_i X_i] \leq \mu_2$. Then

$$\Pr[\sum_i a_i X_i \leq (1 - \delta)\mu_1] \leq e^{-\mu_1 \delta^2/2}$$

$$\Pr[\sum_i a_i X_i \geq (1 + \delta)\mu_2] \leq \Big(\frac{e^\delta}{(1+\delta)^{1+\delta}}\Big)^{\mu_2}$$

Proof. See Gandhi et al. [18, Theorem 3.1]. ■

We will also use the following corollary.

Corollary 4.3. *Assume the same hypotheses as Theorem 4.2. Let $\mu = E[\sum_i a_i X_i]$. Then*

$$\Pr[|\sum_i a_i X_i - \mu| \geq \delta\mu] \leq 2e^{-\frac{\delta^2 \mu}{2(1+\delta/3)}}.$$

Proof. Follows from Theorem 4.2 and basic calculus. See also McDiarmid [34, Theorem 2.3]. ■

Now consider the approach of Algorithm 2 for constructing a sparsifier. In each round of sampling, instead of picking edges independently, we pick a uniformly random spanning tree. Every edge e in the tree is assigned weight c_e. This sampling is repeated for ρ rounds, and the sparsifier is the sum of these weighted trees.

By Lemma B.2, the probability of sampling any particular edge is the same as when sampling by effective conductances, as was done in Corollary 1.3. Furthermore, the same analysis as Section 2 shows that this sampling method also produces a sparsifier — the only change to the analysis is that all uses of Chernoff bounds (namely, in Claims 2.1, 2.2 and 2.3) can be replaced with the concentration bounds in Theorem 4.2 and Corollary 4.3. This completes the proof of Theorem 1.5.

4.1 Lower bound on number of trees

In this section, we consider the tradeoff between the number of trees (i.e., the value ρ) and the quality of sparsification in Theorem 1.5. We prove a lower bound on the number of trees necessary to produce a sparsifier with a given approximation factor.

Proof (of Lemma 1.10). Let G be a graph defined as follows. Its vertices are $\{u_1, \ldots, u_n\} \cup \{v_1, \ldots, v_{n+1}\}$. For every $i = 1, \ldots, n$, add k parallel edges $v_i v_{i+1}^{(1)}, \ldots, v_i v_{i+1}^{(k)}$, and a single length-two path v_i-u_i-v_{i+1}. The edges $v_i v_{i+1}^{(j)}$ are called *heavy*, and the edges $v_i u_i$ and $u_i v_{i+1}$ are called *light*. Note that the heavy edges each have effective conductance exactly $(2k+1)/2$. The light edges each have effective conductance exactly $(2k+1)/(k+1) < 2$.

A uniform random spanning tree in this graph can be constructed by repeating the following experiment independently for each $i = 1, \ldots, n$. With probability $2k/(2k+1)$, add a uniformly selected heavy edge $v_i v_{i+1}^{(j)}$ to the tree, and add a uniformly selected light edge $v_i u_i$ or $u_i v_{i+1}$ to the tree. In this case we say that the tree is "heavy in position i". Otherwise, with probability $1/(2k+1)$, add both light edges $v_i u_i$ and $u_i v_{i+1}$ to the tree but no heavy edges. In this case we say that the tree is "light in position i".

Our sampling procedure produces a sparsifier that is the union of ρ trees, where every edge e in the sparsifier is assigned weight c_e/ρ. Suppose there is an i such that all sampled trees are light in position i. Then the cut defined by vertices $\{v_1, u_1, v_2, u_2, \ldots, v_i\}$ has weight exactly $(2k+1)/(k+1) < 2$ in the sparsifier, whereas the true value of the cut is $k+1$.

The probability that at least one tree is heavy in position i is $1 - (2k+1)^{-\rho}$. The probability that there exists an i such that every tree is light in position i is

$$p \;:=\; 1 - (1 - (2k+1)^{-\rho})^n$$

Suppose $\rho = \ln n / \ln(2k+1)$. Then $\lim_{n\to\infty} p = 1 - 1/e$. So with constant probability, there is an i such that every tree is light in position i, and so the sparsifier does not approximate the original graph better than a factor $\frac{k+1}{2}$. ■

References

[1] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 227–236, 2007.

[2] Joshua D. Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-Ramanujan sparsifiers. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing*, pages 255–262, 2009.

[3] András A. Benczúr. *Cut structures and randomized algorithms in edge-connectivity problems.* PhD thesis, Massachusetts Institute of Technology, 1997.

[4] András A. Benczúr and David R. Karger. Approximate *s-t* min-cuts in $\tilde{O}(n^2)$ time. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC)*, 1996.

[5] András A. Benczúr and David R. Karger. Randomized approximation schemes for cuts and flows in capacitated graphs, 2002. arXiv:cs/0207078.

[6] Béla Bollobás. *Modern Graph Theory.* Springer, 1998.

[7] R. L. Brooks, C. A. B. Smith, A. H. Stone, and W. T. Tutte. The dissection of rectangles into squares. *Duke Math. J.*, 7:312–340, 1940.

[8] Charles J. Colbourn, Wendy J. Myrvold, and Eugene Neufeld. Two algorithms for unranking arborescences. *J. Algorithms*, 20(2):268–281, 1996.

[9] Luc Devroye. *Non-Uniform Random Variate Generation*. Springer, 1986. Available at `http://cg.scs.carleton.ca/~luc/rnbookindex.html`.

[10] E. A. Dinic, A. V. Karzanov, and M. V. Lomonosov. The structure of a system of minimal edge cuts of a graph. In A. A. Fridman, editor, *Studies in Discrete Optimization*, pages 290–306. Izdatel'stvo "Nauka", 1976.

[11] Yefim Dinitz and Alek Vainshtein. The connectivity carcass of a vertex subset in a graph and its incremental maintenance. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC)*, pages 716–725, 1994.

[12] Yefim Dinitz and Alek Vainshtein. The general structure of edge-connectivity of a vertex subset in a graph and its incremental maintenance. odd case. *SIAM Journal on Computing*, 30(3):753–808, 2000.

[13] Peter G. Doyle and J. Laurie Snell. Random walks and electric networks. arXiv:math.PR/0001057.

[14] T. Feder and M. Mihail. Balanced matroids. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing (STOC)*, 1992.

[15] Tamás Fleiner and András Frank. A quick proof for the cactus representation of mincuts. EGRES Quick-Proof No. 2009-03, Egerváry Research Group on Combinatorial Optimization, Eötvös Loránd University, 2009.

[16] Tamás Fleiner and Tibor Jordán. Coverings and structure of crossing families. *Mathematical Programming*, 84(3):505–518, 1999.

[17] R. M. Foster. The average impedance of an electric network. In *Contributions to Applied Mechanics*, pages 333–340. Edwards Bros., 1949.

[18] R. Gandhi, S. Khuller, S. Parthasarathy, and A. Srinivasan. Dependent Rounding and its Applications to Approximation Algorithms. *Journal of the ACM*, 53:324–360, 2006.

[19] Navin Goyal, Luis Rademacher, and Santosh Vempala. Expanders via random spanning trees. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 576–585, 2009.

[20] Ramesh Hariharan and Debmalya Panigrahi. A general framework for graph sparsification, April 2010. `http://arxiv.org/abs/1004.4080`.

[21] Toshihide Ibaraki. Computing edge-connectivity in multiple and capacitated graphs. In *SIGAL '90: Proceedings of the international symposium on Algorithms*, pages 12–20, New York, NY, USA, 1990. Springer-Verlag New York, Inc.

[22] David R. Karger. Global min-cuts in RNC, and other ramifications of a simple min-cut algorithm. In *Proceedings of the 4th Annual ACM/SIAM Symposium on Discrete Algorithms (SODA)*, pages 21–30, 1993.

[23] David R. Karger. Random sampling in cut, flow, and network design problems. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing (STOC)*, 1994.

[24] David R. Karger. Random sampling in cut, flow, and network design problems. *Mathematics of Operations Research*, 24(2):383–413, May 1999.

[25] David R. Karger and Clifford Stein. A new approach to the minimum cut problem. *Journal of the ACM*, 43(4):601–640, July 1996.

[26] Jonathan A. Kelner and Aleksander Madry. Faster generation of random spanning trees. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 13–21, 2009.

[27] Rohit Khandekar, Satish Rao, and Umesh V. Vazirani. Graph partitioning using single commodity flows. *Journal of the ACM*, 56(4), 2009.

[28] G. Kirchhoff. Über die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird. *Ann. Phys. und Chem.*, 72:497–508, 1847.

[29] D.E. Knuth and A.C. Yao. The complexity of nonuniform random number generation. In J. E. Traub, editor, *Algorithms and Complexity*, pages 357–428. Academic Press, 1976.

[30] Ioannis Koutis, Gary L. Miller, and Richard Peng. Approaching optimality for solving SDD systems, 2010. arXiv:1003.2958.

[31] László Lovász. Random walks on graphs: A survey. In D. Miklós, V. T. Sós, and T. Szönyi, editors, *Combinatorics, Paul Erdos is Eighty, Volume 2*, pages 353–398. János Bolyai Mathematical Society, 1996.

[32] Russell Lyons and Yuval Peres. *Probability on Trees and Networks*. Cambridge University Press, 2010. In preparation. Current version available at `http://mypage.iu.edu/~rdlyons/`.

[33] Wolfgang Mader. A reduction method for edge-connectivity in graphs. *Ann. Discrete Math.*, 3:145–164, 1978.

[34] Colin McDiarmid. Concentration. In M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 1–46. Springer, 1998.

[35] Hiroshi Nagamochi and Toshihide Ibaraki. A linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph. *Algorithmica*, 7(5&6):583–596, 1992.

[36] Daniel A. Spielman. Algorithms, graph theory, and linear equations. In *Proceedings of the International Congress of Mathematicians*, 2010.

[37] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 563–568, 2008.

[38] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC)*, pages 81–90, 2004. `http://arxiv.org/abs/cs.DS/0310051`.

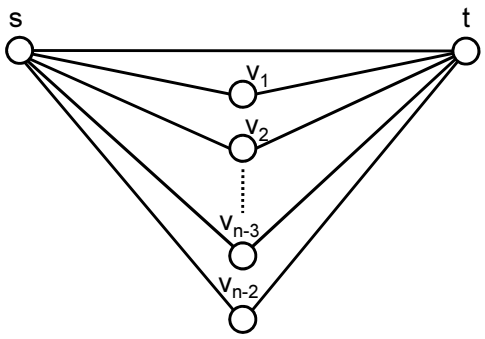

Figure 1: Example showing that conductance can be $\Omega(n)$ times larger than strength

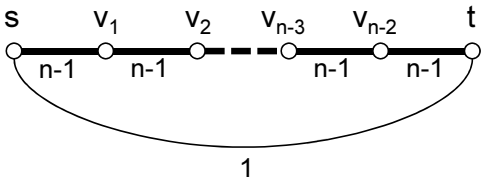

Figure 2: Example showing that strength can be $\Omega(n)$ times larger than conductance

[39] Daniel A. Spielman and Shang-Hua Teng. Spectral sparsification of graphs, 2008. `http://arxiv.org/abs/0808.4134`.

[40] Prasad Tetali. Random walks and the effective resistance of networks. *Journal of Theoretical Probability*, 4(1):101–109, 1991.

A Discussion of k_{st}, c_{st} and k'_{st}

As mentioned in the introduction, the three quantities of an edge st that we consider (edge connectivity, effective conductance and edge strength) all roughly measure the connectivity between s and t. However their values can differ significantly. In this section, we illustrate this with some examples.

Consider a graph which consists of exactly one edge st. To increase k_{st} by k, we can simply add k edge disjoint paths between s and t. In the following examples, we can see that no matter how large k is, it is possible that k'_{st} or c_{st} increases only by one while the other increases by $\Omega(k)$.

- In Figure 1, s and t are connected by an edge st and $n-2$ paths of length 2. Clearly $k_{st} = n-1$, $c_{st} = \frac{1}{2}(n-2)+1 = n/2$. But $k'_{st} = 2$ as every induced subgraph with at least two vertices is at most 2 edge connected.
- In Figure 2, s and t are connected by an edge st and a path of length $n-1$ which consists of edges of weight $n-1$. The graph is n-edge-connected so $k_{st} = k'_{st} = n$ but $c_{st} = \frac{n-1}{n-1} + 1 = 2$.

Although c_{st} and k'_{st} are incomparable, they are upper bounded by k_{st}.

Claim A.1. *For any edge $st \in E$, $k_{st} \geq \max\{c_{st}, k'_{st}\}$.*

Proof. It is immediate from the definition of edge strength that $k'_{st} \leq k_{st}$, so we focus on the effective conductance. Since the connectivity between s and t is k_{st}, there is a cut of size k_{st}

separating s and t. Contracting both sides of the cut, we get two new vertices s' and t'. By Rayleigh monotonicity [13], $c_{s't'}$ is at least c_{st}. Clearly $c_{s't'} = k_{st}$, so the proof is complete. ■

B Corollaries of Theorem 1.1

First we show that our corollaries satisfy the hypotheses of Theorem 1.1. By Claim A.1, Corollaries 1.2, 1.3 and 1.4 all have $\kappa_e \leq k_e$, so Theorem 1.1 is applicable.

It remains to analyze $|E'|$, the number of sampled edges. For Corollaries 1.2 and 1.4 we use a property of edge strength proved by Benczúr and Karger [5, Lemma 2.7].

Lemma B.1. *In a multigraph with edge strengths k'_e, we have*

$$\sum_{e \in E} 1/k'_e \;\leq\; n-1.$$

Here the sum is over all copies of the multiedges.

Thus, for Corollaries 1.2 and 1.4, we have

$$\mathrm{E}\left[\,|E'|\,\right] \;=\; \rho \sum_{e \in E} p_e \;=\; \rho \sum_{e \in E} 1/\kappa_e \;\leq\; \rho \sum_{e \in E} 1/k'_e \;\leq\; \rho(n-1) \;=\; O(n \log^2 n/\epsilon^2).$$

Finally, we must bound the size of E' in Corollary 1.3. We require the following lemma.

Lemma B.2. *Let $G = (V, E)$ be a multigraph, and let T be a spanning tree in G chosen uniformly at random. Then, for any copy of an edge $e \in E$, $\Pr[e \in T] = 1/c_e$.*

Proof. See Kirchhoff [28], Brooks et al. [7, pp. 318], Lovász [31, Theorem 4.1(i) and Corollary 4.2] and Lyons and Peres [32, Corollary 4.4]. ■

This immediately implies that

$$\sum_{e \in E} 1/c_e \;=\; \sum_{e \in E} \Pr\left[\, e \in T \,\right] \;=\; \mathrm{E}\left[\,|E(T)|\,\right] = n-1.$$

This fact is known as Foster's theorem, and it is independently due to Foster [17] and Tetali [40]. Thus,

$$\mathrm{E}\left[\,|E'|\,\right] \;=\; \rho \sum_{e \in E} p_e \;=\; \rho \sum_{e \in E} 1/c_e \;=\; \rho(n-1) \;=\; O(n \log^2 n/\epsilon^2).$$

C Motivation for Partitioning Edges

The natural first approach to proving Theorem 1.1 would be to bound the probability of large deviation for each cut and then union bound over all cuts. This approach is not feasible, as can be illustrated using the example in Figure 3.

In this graph, s and t are connected by k parallel edges and k paths of length 2. Recall that in our sampling scheme each copy of st is sampled with probability $1/2k$ for $\rho = O(\log^2 n)$ rounds and is assigned a weight of $2k/\rho$ if sampled. Each edge other than st is sampled with probability $1/2$ for ρ rounds and is assigned a weight of $2/\rho$ if sampled. Consider a set $U \subseteq V - t$ that contains s. Then $|\delta(U)|$ is at most $2k$. Suppose we want to bound the probability that the sampled weight $w(\delta(U))$ exceeds $4k$. For this to happen, at most 2ρ copies of st can be included in the sparsifier.

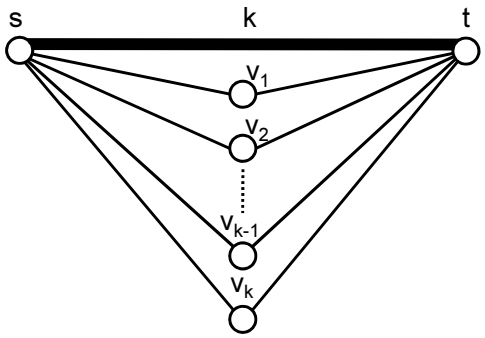

Figure 3: The cut-induced set consisting of st is overcounted 2^n times if we simply union bound over all cuts.

By a Chernoff bound, this many edges are included with probability at most $e^{-2\rho} = e^{O(-\log^2 n)}$. However, there are $\Omega(2^n)$ such U's, which is too many for such a union bound to work.

The reason this union bound fails is that the event "more than 2ρ copies of st are included" is overcounted $\Theta(2^n)$ times, once for each U. However since all $\delta(U)$'s share the same k copies of st, we actually only need to analyze this event once.

Benczúr and Karger [5] accomplished this by decomposing the graph. Assume that the edges $E = \{e_1, \ldots, e_m\}$ are sorted by increasing edge strengths. Each G_j contains all edges e_i with $i \geq j$. Then G can be viewed as the sum of G_j's, with each G_j scaled by $k'_{e_j} - k'_{e_{j-1}}$. An important property of this decomposition is that if e_i is in G_j then the strength of e_i in G_j is the same as its in the original graph G. This is because the k'_{e_i}-strong component H in G that contains e_i must also be present in G_j, as all edges in H have strengths at least k'_{e_i}.

Therefore, even though edges in G_j have small sampling probabilities (at most $1/k'_{e_j}$), the expected number of sampled edges in every cut is at least $\Omega(\log n/\epsilon^2)$, since the min cut of G_j is large (at least k'_{e_j}). Thus Karger's graph skeleton analysis is applicable to sampling in G_j. Roughly speaking, in order to use the Chernoff bound to obtain a constant factor approximation in the number of sampled edges in a cut with a failure probability of $n^{-\Omega(1)}$, the expected number of sampled edges in the cut needs to be $\Omega(\log n)$.

To prove Theorem 1.1, we could attempt to use the same decomposition to analyze our sampling scheme where edge connectivity is used instead of strength. The problem is that in general edge connectivity is not preserved under such decomposition. To see this, consider the example in Figure 1. Observe that the subgraph induced by those edges with connectivities at least $n-1$ consists of only one edge st, so this subgraph has min cut value 1. The expected number of copies of st in the sparsifier is $O(\log^2 n/n)$, so we cannot expect to say that sampling preserves every cut of this subgraph to within $1 \pm \epsilon$.

D Proofs for Section 2.1

In this section we prove Claim 2.1, Claim 2.2 and Claim 2.3. We require the following three versions of the Chernoff bound. For the case $U = 1$, these can be found in the survey of McDiarmid [34]; the case of larger U reduces to that case by scaling.

Theorem D.1. *Let $X_1, \ldots, X_k$ be independent random variables with values in $[0,1]$. Let $\alpha_1, \ldots, \alpha_k$ be scalars in $[0, U]$. Let X be a weighted sum of Bernoulli trials defined by $X =$*

$\sum_{i=1}^{k} \alpha_i X_i$, *and let* $\mu = \mathrm{E}[X]$. *Then for any* $\delta > 0$, *we have*

$$\Pr[|X - \mu| \geq \delta\mu] \leq 2\exp\left(-\frac{\delta^2\mu}{2(1+\delta/3)U}\right).$$

Corollary D.2. *Let* X *and* μ *be as in Theorem D.1. Then for any* $0 < \delta < 1$, *we have*

$$\Pr[|X - \mu| \geq \delta\mu] \leq 2e^{-\delta^2\mu/3U}.$$

Theorem D.3. *Let* X *and* μ *be as in Theorem D.1. Then for any* $\delta > 0$, *we have*

$$\Pr[X - \mu \geq \delta\mu] \leq \exp\left(-\frac{g(\delta)\mu}{U}\right),$$

where $g(\delta) = (1+\delta)\ln(1+\delta) - \delta$ *is the function defined in Eq.* (2.2).

Claim 2.1. *Let* $F \subseteq E_i$ *be a cut-induced set with* $q(F) \leq |F| \ln n$. *Then*

$$\Pr[\mathcal{A}_F] \leq 2n^{-d\alpha(F)\epsilon^2/6}.$$

Proof. Let $U = \frac{2^{i+1}}{\rho}$ and $\mu = \mathrm{E}[X_F] = |F|$. By the definition of our sampling process, X_F is a weighted sum of Bernoulli trials where each weight is less than U. Thus

$$\begin{aligned}
\Pr[\mathcal{A}_F] &= \Pr[|X_F - |F|| > \epsilon|F|] \\
&\leq 2\exp\left(-\frac{\epsilon^2|F|}{3U}\right) \qquad \text{(by Corollary D.2)} \\
&\leq 2\exp\left(-\frac{\epsilon^2 q(F)\rho}{3\ln(n)2^{i+1}}\right) \qquad \text{(by our assumption on } q(F)) \\
&\leq 2\exp\left(-\frac{\epsilon^2\alpha(F)d\ln n}{6}\right) \\
&\leq 2n^{-d\alpha(F)\epsilon^2/6}.
\end{aligned}$$

This concludes the proof. ■

Claim 2.2. *Let* $F \subseteq E_i$ *be a cut-induced set with* $q(F) > |F| \ln n$. *Then*

$$\Pr[\mathcal{B}_F] \leq 2n^{-d\alpha(F)\epsilon^2/8}.$$

Proof. Let $U = \frac{2^{i+1}}{\rho}$, $\mu = \mathrm{E}[X_F] = |F|$ and $\delta = \frac{\epsilon q(F)}{\ln(n)|F|}$. Then X_F is a weighted sum of Bernoulli trials, and U is an upper bound on the weights. Note that $\delta \geq \epsilon$ and $1 \geq \epsilon$. Thus

$$\frac{\delta^2}{1+\delta/3} \geq \frac{\delta\epsilon}{1+\epsilon/3} \geq \frac{\delta\epsilon}{2}.$$

Also,

$$\frac{\delta\mu}{U} = \frac{\epsilon q(F)\rho}{\ln(n)2^{i+1}} = \frac{\epsilon\alpha(F)d\ln n}{2}. \tag{D.1}$$

Thus

$$\begin{aligned}
\Pr[\,\mathcal{B}_F\,] &= \Pr\left[\,|X_F - |F|| \geq \frac{\epsilon q(F)}{\ln n}\,\right] \\
&= \Pr[\,|X_F - |F|| \geq \delta|F|\,] \\
&\leq 2\exp\left(-\frac{\delta^2\mu}{2(1+\delta/3)U}\right) \qquad \text{(by Theorem D.1)} \\
&\leq 2\exp\left(-\frac{\delta\epsilon\mu}{4U}\right) \\
&\leq 2\exp\left(-\frac{\epsilon^2\alpha(F)d\ln n}{8}\right) \qquad \text{(by Eq. (D.1))} \\
&= 2n^{-d\alpha(F)\epsilon^2/8}.
\end{aligned}$$

This concludes the proof. ■

Claim 2.3. *For every cut-induced set F,*

$$\Pr[\,\mathcal{C}_F\,] \leq n^{-d\alpha(F)\epsilon/2}.$$

Proof. Let $U = \frac{2^{i+1}}{\rho}$ and $\mu = \mathrm{E}[\,X_F\,] = |F|$. Then

$$\begin{aligned}
\Pr[\,\mathcal{C}_F\,] &= \Pr\left[\,X_F - |F| > g^{-1}\Big(\frac{\epsilon^2 q(F)}{|F|\ln n}\Big)\cdot|F|\,\right] \\
&\leq \exp\left(-\Big(\frac{\epsilon^2 q(F)}{|F|\ln n}\Big)\frac{|F|}{U}\right) \qquad \text{(by Theorem D.3)} \\
&\leq \exp\left(-\frac{\epsilon^2 q(F)\rho}{\ln(n)2^{i+1}}\right) \\
&\leq \exp\left(-\frac{\epsilon^2\alpha(F)d\ln n}{2}\right).
\end{aligned}$$

This completes the proof. ■

Claim 2.4. *By choosing $d = O(1/\epsilon^2)$ sufficiently large, then with high probability, every cut-induced set F satisfies*

- *if $q(F) \leq |F|\ln n$ then $\mathcal{A}_F$ does not hold;*
- *if $q(F) > |F|\ln n$ then $\mathcal{B}_F$ does not hold; and*
- *$\mathcal{C}_F$ does not hold.*

Proof. Fix an i and let $F^1, F^2, \ldots$ be all the cut-induced subsets of E_i, ordered such that $q(F^1) \leq q(F^2) \leq \ldots$. Let

$$p_j = \begin{cases} \Pr[\,\mathcal{A}_F \cup \mathcal{C}_F\,] & (\text{if } q(F) \leq |F|\ln n) \\ \Pr[\,\mathcal{B}_F \cup \mathcal{C}_F\,] & (\text{if } q(F) > |F|\ln n). \end{cases}$$

By Claims 2.1, 2.2 and 2.3, there exists a value $d = O(1/\epsilon^2)$ such that

$$p_j \;\le\; 4n^{-(r+2)\alpha(F^j)}. \tag{D.2}$$

We consider the first n^2 cut-induced sets. Note that for all $j \ge 1$, $p_j \le 4n^{-(r+2)}$. Therefore, a union bound shows that the probability that any bad event happens for some F^j with $1 \le j \le n^2$ is at most $n^2 \cdot p_1 \le 4n^{-r}$.

Now we consider the remaining cut-induced sets $F^j \subseteq E_i$ for $j > n^2$. Corollary 2.5 states that, for any $\alpha \ge 1$,

$$|\{\text{ cut-induced set } F \subseteq E_i \,:\, \alpha(F) \le \alpha\}| \;<\; n^{2\alpha}.$$

Letting $\alpha > 1$ be such that $j = n^{2\alpha}$, we see that $\alpha(F^j) > \alpha = \frac{\ln j}{2\ln n}$. Thus, from Eq. (D.2) we have

$$p_j \;\le\; 4n^{-(r+2)\alpha(F^j)} \;<\; 4n^{-(r+2)\ln(j)/2\ln n} \;\le\; 4j^{-(r+2)/2}$$

Thus

$$\begin{aligned}
\sum_{j>n^2} p_j \;&\le\; \sum_{j>n^2} 4j^{-(r+2)/2} \\
&\le\; \int_{n^2}^{\infty} 4j^{-(r+2)/2}\,dj \\
&=\; \left.\frac{-8j^{-r/2}}{r}\right|_{n^2}^{\infty} \\
&=\; O(n^{-r}).
\end{aligned}$$

This completes the proof. ■

E Proofs for Section 2.2

Claim 2.7. *Let $F \subseteq E_i$ be a cut-induced set. Then $|F| < n^2 2^i$.*

Proof. Since $F \subseteq E_i$, every $e \in F$ satisfies $k_e \le 2^{i+1}$. Since $u_e \le k_e$ for every e, we obtain $u_e \le 2^{i+1}$. Thus $|F| = \sum_{e\in F} u_e \le \binom{n}{2} 2^{i+1}$. This proves the claim. ■

Claim 2.8. *For any integer $d \ge 0$,*

$$\sum_{i \le \lg|C| - 2\lg n - d} |C \cap E_i| \;<\; 2^{1-d}\,|C|.$$

Proof. By Claim 2.7,

$$\sum_{i \le \lg|C| - 2\lg n - d} |C \cap E_i| \;<\; \sum_{i \le \lg|C| - 2\lg n - d} n^2 2^i \;\le\; n^2 2^{\lg|C| - 2\lg n - d + 1} \;=\; 2^{1-d}\,|C|.$$

This completes the proof. ■

Claim 2.9. *Define $h : \mathbb{R} \to \mathbb{R}$ by*

$$h(x) \;=\; \frac{2x}{\ln(1+\sqrt{x})}.$$

Then $g^{-1}(x) \le h(x)$ for all $x \ge 0$.

The purpose of this claim is to give a simple, asymptotically tight upper bound on g^{-1}. We thank "mathphysicist" from the web site MathOverflow for pointing out that a precise expression for g^{-1} can be given using the Lambert W function. Specifically, one can show that

$$g^{-1}(x) \;=\; \exp\Bigl(W\bigl(\tfrac{x-1}{e}\bigr)+1\Bigr) - 1.$$

Unfortunately this exact expression is not terribly useful, since we do not know of any simple, asymptotically tight bounds on W.

Proof (of Claim 2.9). For all $x \ge 0$, we have

$$\begin{aligned}
& & \sqrt{x} \;&\ge\; \ln(1+\sqrt{x}) \\
&\Longrightarrow & h(x) \;&=\; \frac{2x}{\log(1+\sqrt{x})} \;\ge\; \sqrt{x} \\
&\Longrightarrow & \ln(1+h(x)) \;&\ge\; \ln(1+\sqrt{x}) \\
&\Longrightarrow & h(x)\cdot\ln(1+h(x)) \;&=\; \frac{2x\cdot\ln(1+h(x))}{\ln(1+\sqrt{x})} \;\ge\; 2x. \qquad \text{(E.1)}
\end{aligned}$$

Next, for $y \ge 0$,

$$\frac{1}{1+y} \;\ge\; \frac{1}{1+y} - \frac{y}{(1+y)^2}.$$

Integrating, we obtain

$$\begin{aligned}
& \ln(1+y) \;\ge\; \frac{y}{1+y} \\
\Longrightarrow \qquad & \ln(1+y) \;\ge\; \frac{1}{2}\Bigl(\ln(1+y) + \frac{y}{1+y}\Bigr).
\end{aligned}$$

Integrating again, we obtain

$$(1+y)\ln(1+y) - y \;\ge\; \frac{y\ln(1+y)}{2}.$$

Substituting $y = h(x)$ into this and applying Eq. (E.1) yields

$$g(h(x)) \;=\; (1+h(x))\ln(1+h(x)) - h(x) \;\ge\; \frac{h(x)\ln(1+h(x))}{2} \;\ge\; x.$$

As observed above, g^{-1} is strictly monotonically increasing. Thus $g(h(x)) \ge x$ implies that $h(x) \ge g^{-1}(x)$. ■

F Probability of success in Algorithm 3

In this appendix we complete the proof of Theorem 3.3 by proving the following claim.

Claim 3.7. *The probability that Algorithm 3 outputs F is at least $n^{-2\alpha}$.*

Algorithm 4 An algorithm for finding a small cut-induced set by contracting random walks.

procedure ContractRW($G,\ B,\ \alpha$)

input: A graph $G = (V, E)$, a set $B \subseteq E$, and an approximation factor α

output: A cut-induced subset of B

While there are more than $\lceil 2\alpha \rceil$ black vertices remaining

Randomly pick a black vertex u_1 with probability proportional to its degree

Perform a random walk starting from u_1 and stopping when it hits a black vertex u_2 (possibly $u_1 = u_2$)

If $u_1 \neq u_2$

Contract all edges traversed by the random walk and remove any self loops

Pick a non-empty proper subset S of V uniformly at random and output the black edges with exactly one endpoint in S

Proof. Define $R := \lceil 2\alpha \rceil$. In the last iteration of the algorithm, the number of remaining vertices is at least $R + 1$. The probability that the invariants hold at the end of the algorithm is at least the product of the probabilities that the invariants are not violated at any step of the algorithm. By Claims 3.5 and 3.6, this probability is at least

$$\prod_{r=n}^{R+1} 1 - \frac{2\alpha}{r} \quad = \quad \frac{n-2\alpha}{n} \cdot \frac{n-1-2\alpha}{n-1} \cdots \frac{R+1-2\alpha}{R+1} \quad = \quad \frac{(n-2\alpha)!}{(R-2\alpha)!} \frac{R!}{n!},$$

where the factorial function is extended to arbitrary real numbers via the Gamma function.

Since there are at most R remaining vertices at the end of the algorithm there are less than 2^{R-1} remaining non-trivial cuts, at least one of which induces F, by (I1). Therefore, the probability that the last step of the algorithm selects a set S that induces F is at least

$$2^{1-R} \frac{(n-2\alpha)!}{(R-2\alpha)!} \frac{R!}{n!} \ > \ \frac{1}{n^{2\alpha}} \frac{1}{(R-2\alpha)!} \ \geq \ n^{-2\alpha},$$

where we have used the inequalities $2^{R-1} \leq R!$, $n!/(n-2\alpha)! < n^{2\alpha}$, and $x! \leq 1$ for $x \in [0, 1]$. ■

G Random contraction algorithm by random walks

In this appendix we present Algorithm 4, which is a variant of the contraction algorithm that contracts random walks instead of random edges. We use the similar terminology and notation to Section 3, e.g., black vertices. The following analysis of the algorithm immediately implies Theorem 3.2.

Theorem G.1. *For any cut-induced set $F \subseteq B$ with $q(F) \leq \alpha K$, Algorithm 3 outputs F with probability at least $n^{-2\alpha}$.*

The approach for proving Theorem G.1 is again similar to Karger's analysis of the contraction algorithm — for any cut C, we can bound the probability that an edge in C is contracted. Formally, our analysis is:

Lemma G.2. *Consider an iteration of the while loop that begins with r remaining black vertices. Suppose that no edge in C has been contracted so far. Suppose that the random walk in this iteration*

has $u_1 \neq u_2$. Then

$$\Pr[\text{some edge in } C \text{ is contracted in this iteration}] \;\leq\; \frac{2|C|}{Kr}.$$

To prove Theorem G.1, one applies Lemma G.2 where C is a cut which induces F and satisfies $|C| \leq q(F) \leq \alpha K$. The remainder of the proof follows by the same argument as Claim 3.7.

The key method in proving Lemma G.2 is to understand the probability that a random walk hits a certain set of vertices before hitting some other set of vertices. To that end, let us introduce some notation. For any two sets of vertices X and Y, let $\mathrm{cond}_{X,Y}$ denote the effective conductance between X and Y. Equivalently, identify X into a single vertex x, identify Y into a single vertex y, and let $\mathrm{cond}_{X,Y}$ be the effective conductance between x and y.

Next, suppose that $s \in V$ and that T and U are disjoint subsets of V. We use $s \to T < U$ to denote the event that a random walk starting at s hits T before it hits U. If $T = \{t\}$, we use the shorthand $s \to t < U$, and similarly if $U = \{u\}$. In the case that $s \in T \cup U$, the term "hits" means "hits after performing at least one step of the random walk".

Remark. *The event $s \to T < U$ can also be understood in another way. Let G' be the graph obtained by identifying all nodes in T into a single node t, and identifying all nodes in U into a single node u. Then $\Pr[s \to T < U]$ equals the probability that a random walk in G' starting at s hits t before u.*

The main tool in the proof of Lemma G.2 is the following reciprocity law. It will allow us to consider random walks originating at the cut C rather than random walks that cross C.

Lemma G.3. *Let $s, t \in V$ and $U \subseteq V$. Assume that $s \neq t$ and $\{s,t\} \cap U = \emptyset$. Then*

$$\mathrm{cond}_{s,\{t\}\cup U} \cdot \Pr[s \to t < U] \;=\; \mathrm{cond}_{t,\{s\}\cup U} \cdot \Pr[t \to s < U]. \tag{G.1}$$

To prove this lemma, we need to understand the relationship between the following events:

$$\begin{aligned} \mathcal{E} &= s \to T < \{s\} \cup U \\ \mathcal{E}' &= s \to T \cup U < s \\ \mathcal{E}^* &= s \to T < U. \end{aligned}$$

In English, $\mathcal{E}$ is the event that the random walk hits T before hitting U or returning to s, $\mathcal{E}'$ is the event that the random walk hits T or U before returning to s, and $\mathcal{E}^*$ is the same as $\mathcal{E}$ except that the random walk is permitted to return to s before hitting T or U.

Claim G.4. $\mathcal{E} = \mathcal{E}^* \wedge \mathcal{E}'$.

Claim G.5. *$\mathcal{E}^*$ and $\mathcal{E}'$ are independent.*

Proof. The claim essentially follows from the "craps principle". In more detail, consider any random walk starting s and ending at $T \cup U$. It can be viewed as a sequence $w_1, \ldots, w_k$ of random walks where

- for each $i < k$, w_i starts and ends at s but otherwise does not traverse s,
- w_k starts at s and ends at $T \cup U$ but otherwise does not traverse s.

So $\Pr[\mathcal{E}^*]$ is the probability that w_k ends at T, and by the Markov property, this equals

$$\Pr[s \to T < U \mid s \to T \cup U < s].$$

Thus we have argued that $\Pr[\mathcal{E}^*] = \Pr[\mathcal{E}^* \mid \mathcal{E}']$, as required. ■

Claim G.6. $\Pr[\mathcal{E}'] = \mathrm{cond}_{s,T\cup U}/d(s)$.

Proof. Doyle and Snell [13, §1.3.4]. ■

Proof (of Lemma G.3). It is known [6, Theorem IX.22] that

$$d(s)\cdot\Pr[s\to t<\{s\}\cup U] \;=\; d(t)\cdot\Pr[t\to s<\{t\}\cup U].$$

By Claim G.4 and Claim G.5, this is equivalent to

$$d(s)\cdot\Pr[s\to\{t\}\cup U<s]\cdot\Pr[s\to t<U] \;=\; d(t)\cdot\Pr[t\to\{s\}\cup U<t]\cdot\Pr[t\to s<U].$$

By Claim G.6, this is equivalent to (G.1). ■

Proof (of Lemma G.2). It is more convenient to consider hitting a vertex than a cut, so we subdivide every edge in C and merge the subdividing vertices into a new vertex z. We consider the random walk in the modified graph induced by the random walk in the original graph. The former walk hits z iff the latter walk intersects C.

For the remainder of this proof, W denotes the set of currently remaining black vertices.

Claim G.7. *For any $w\in W$,*

$$\Pr[u_1=w\mid u_1\neq u_2] \;=\; \frac{\mathrm{cond}_{w,W\setminus\{w\}}}{\sum_{u\in W}\mathrm{cond}_{u,W\setminus\{u\}}}.$$

Proof. Let $D=\sum_{v\in W}d(v)$. Then

$$\begin{aligned}
\Pr[u_1=w\mid u_1\neq u_2] &= \frac{\Pr[u_1=w\ \wedge\ u_1\neq u_2]}{\sum_{u\in W}\Pr[u_1=u\ \wedge\ u_1\neq u_2]}\\
&= \frac{\Pr[u_1=w]\cdot\Pr[u_1\neq u_2\mid u_1=w]}{\sum_{u\in W}\Pr[u_1=u]\cdot\Pr[u_1\neq u_2\mid u_1=u]}\\
&= \frac{\big(d(w)/D\big)\cdot\Pr[w\to W\setminus\{w\}<w]}{\sum_{u\in W}\big(d(u)/D\big)\cdot\Pr[u\to W\setminus\{u\}<u]}\\
&= \frac{d(w)\cdot\big(\mathrm{cond}_{w,W\setminus\{w\}}/d(w)\big)}{\sum_{u\in W}d(u)\cdot\big(\mathrm{cond}_{u,W\setminus\{u\}}/d(u)\big)} \qquad \text{(by Claim G.6)}\\
&= \frac{\mathrm{cond}_{w,W\setminus\{w\}}}{\sum_{u\in W}\mathrm{cond}_{u,W\setminus\{u\}}}.
\end{aligned}$$

This proves the claim. □

Claim G.8. *Let $\mathcal{A}$ and $\mathcal{B}$ be disjoint events. Let $\mathcal{C}$ be another event.*

$$\Pr[\mathcal{C}\mid\mathcal{A}]\geq\Pr[\mathcal{C}\mid\mathcal{A}\vee\mathcal{B}] \quad\Longrightarrow\quad \Pr[\mathcal{C}\mid\mathcal{B}]\leq\Pr[\mathcal{C}\mid\mathcal{A}\vee\mathcal{B}].$$

Claim G.9. *For any $w\in W$,*

$$\Pr[w\to z<W\setminus\{w\}\ \mid\ w\to W\setminus\{w\}<w] \;\leq\; \Pr[w\to z<W\setminus\{w\}].$$

Proof. Define

$$\begin{aligned}\mathcal{B} &= w \to W \setminus \{w\} < w \\ \mathcal{A} &= (w \to z < w) \setminus \mathcal{B} \\ \mathcal{C} &= w \to z < W \setminus \{w\}\,.\end{aligned}$$

Since $\Pr[\mathcal{C} \mid \mathcal{A}] = 1$, the hypotheses of Claim G.8 are satisfied, and therefore

$$\Pr[w \to z < W \setminus \{w\} \mid w \to W \setminus \{w\} < w] \;\le\; \Pr[w \to z < W \setminus \{w\} \mid w \to (W \setminus \{w\}) \cup \{z\} < w].$$

By Claim G.5, the latter quantity equals $\Pr[w \to z < W \setminus \{w\}]$. □

Now, we analyze the probability that the random walk hits the cut. We condition on the event $u_1 \neq u_2$, since that is the only case when the algorithm contracts edges.

$$\begin{aligned}
&\Pr[\text{random walk hits } z \mid u_1 \neq u_2] \\
&= \sum_{w \in W} \Pr[u_1 = w \mid u_1 \neq u_2] \cdot \Pr[\text{ random walk hits } z \;\mid\; u_1 = w \;\wedge\; u_1 \neq u_2] \\
&= \sum_{w \in W} \Pr[u_1 = w \mid u_1 \neq u_2] \cdot \Pr[\, w \to z < W \setminus \{w\} \;\mid\; w \to W \setminus \{w\} < w \,] \\
&= \sum_{w \in W} \frac{\mathrm{cond}_{w, W\setminus\{w\}}}{\sum_{u \in W} \mathrm{cond}_{u, W\setminus\{u\}}} \cdot \Pr[\, w \to z < W \setminus \{w\} \;\mid\; w \to W \setminus \{w\} < w \,] \qquad \text{(by Claim G.7)} \\
&\le \sum_{w \in W} \frac{\mathrm{cond}_{w, W\setminus\{w\}}}{\sum_{u \in W} \mathrm{cond}_{u, W\setminus\{u\}}} \cdot \Pr[w \to z < W \setminus \{w\}] \qquad \text{(by Claim G.9)} \\
&\le \sum_{w \in W} \frac{\mathrm{cond}_{w, \{z\} \cup (W\setminus\{w\})}}{\sum_{u \in W} \mathrm{cond}_{u, W\setminus\{u\}}} \cdot \Pr[w \to z < W \setminus \{w\}] \\
&= \sum_{w \in W} \frac{\mathrm{cond}_{z, W}}{\sum_{u \in W} \mathrm{cond}_{u, W\setminus\{u\}}} \Pr[z \to w < W \setminus \{w\}] \qquad \text{(by Lemma G.3)} \\
&= \frac{\mathrm{cond}_{z, W}}{\sum_{u \in W} \mathrm{cond}_{u, W\setminus\{u\}}} \underbrace{\sum_{w \in W} \Pr[z \to w < W \setminus \{w\}]}_{(=1, \text{ because the walk is ergodic})} \\
&= \frac{\mathrm{cond}_{z, W}}{\sum_{u \in W} \mathrm{cond}_{u, W\setminus\{u\}}} \\
&\le \frac{2|C|}{K|W|}.
\end{aligned}$$

The last inequality is because the node z has degree $2|C|$, and every node $u \in W$ is connected to some node $v \in W \setminus \{u\}$ by a black edge, and $\mathrm{cond}_{u, W\setminus\{u\}} \ge \mathrm{cond}_{u,v} \ge K$. ■

H Algorithms for constructing sparsifiers

In this section, we sketch the algorithms as stated in Theorems 1.7, 1.8 and 1.9. They are simple modifications of the algorithms of Benczúr and Karger [5]. The main difference is that Benczúr and Karger's algorithms compute approximate edge strengths whereas our modifications compute

approximate edge connectivities. The sparsifiers we obtain have slightly larger size but the algorithms are simpler and more efficient because approximating the edge connectivities is quite a bit easier than approximate the edge strengths. Our algorithms can be easily implemented, and furthermore they can be used as a preprocessing step for computing smaller sparsifiers. The proofs of correctness of these algorithms are almost exactly the same as the proofs in [5].

H.1 Finding $O(n \log^3 n)$-size sparsifiers for graphs with polynomial weights

We now present an algorithm that computes a sparsifier of size $O(n \log^3 n)$. It runs in $O(m)$ time for unweighted graphs and $O(m + n \log^5 n)$ time for graphs with polynomial weights. Recall that, in Theorem 1.1, it is sufficient to find κ_e that is a lower bound of the edge connectivity k_e.

The main tool that we use is the *k-certificate* introduced by Nagamochi and Ibaraki [35]. Given a multigraph $G = (V, E)$, they partition E into a set of forests $F_1, F_2, \ldots$ in the follwing way. Let F_1 be a maximal forest of G and for $i > 1$, let F_i be a maximal forest of the subgraph $G - \cup_{j<i} F_j$. Each F_i is called a *NI-forest*. Nagamochi and Ibaraki showed that for any integer $k \geq 0$, $H_k = \cup_{j \leq k} F_j$, the union of the first k NI-forests, preserves all cuts of G that have size at most k. Thus H_k contains all $(k+1)$-light edges of G (an edge e is k-heavy if $k_e \geq k$ and k-light otherwise). H_k is called a k-certificate. Clearly, it contains at most $k(n-1)$ edges.

Nagamochi and Ibaraki [35] presented an algorithm for labeling every edge e with a label r_e, such that if an edge e has multiplicity $u_e > 1$, the u_e copies of e are contained in the u_e NI-forests $F_{r_e - u_e + 1}, \ldots, F_{r_e}$. For simple graphs, it runs in $O(m)$ time. For multigraphs, a slightly modified version [21] of this algorithm runs in $O(m + n \log n)$ time.

Let $e = st$ be an edge. Note that if e appears in F_i, then s and t must be connected in F_j for every $j < i$, for otherwise e can be added to F_j, which contradicts the maximality of F_j. Therefore, r_e is a lower bound of k_e and we can set $\kappa_e = r_e$. Suppose we sample with probability $p_e = 1/\kappa_e$ as described in Theorem 1.1. Then with high probability, this will produce a sparsifier that preserves every cut to within a $1 \pm \epsilon$ factor. Since we assume the weights are polynomially bounded, the expected number of edges per round is

$$\sum_{e \in E} u_e / r_e \leq \sum_{j=1}^{n^2 \operatorname{poly}(n)} \sum_{e \in F_j} 1/j \leq (n-1) \sum_{j=1}^{n^2 \operatorname{poly}(n)} 1/j = O(n \log n).$$

Therefore with high probability, the sparsifier contains $O(n \log^3 n / \epsilon^2)$ edges.

Using the Nagamochi-Ibaraki Certificate algorithm, all κ_e can be found in $O(m + n \log n)$ time. For each edge e, we can decide in expected constant time whether at least one copy of e is included in the sparsifier. For an edge that has at least one copy in the sparsifier, we can find the sum of weights of all copies of it by sampling from the distribution Binomial($u_e \rho$, p_e) instead of sampling u_e Bernoulli random variables per round. This sampling is easy to do in $O(u_e \rho p_e) = O(u_e \log^2 n / k_e \epsilon^2) = O(\log^2 n / \epsilon^2)$ time, for each edge included in the sparsifier. We suspect that this can be improved using the technique of Knuth and Yao [29] [9, Ch. 15], but leave the details to future work. Therefore, the total running time is $O(m + n \log^5 n / \epsilon^4)$ for graphs with polynomial weights.

For unweighted graphs, we can reduce this to $O(m)$ time. Note that for unweighted graphs, $u_e = 1$ for all $e \in E$. If an edge e has $\rho / k_e > 1$, instead of performing ρ rounds of sampling, we can include e with probability 1 and assign it a weight of 1. Thus we can sample the weight of an edge in expected constant time. This change can only decrease the expected size of the sparsifier and the sampled weight of every cut can only be more concentrated around its mean.

Algorithm 5 Algorithm for estimating edge connectivities.

procedure ConnectivityEstimation(H,k)
input: subgraph H of G
$E' \leftarrow$ Partition(H,$2k$)
for each $e \in E'$
    $\kappa_e \leftarrow k$
for each nontrivial connected component $H' \subset H - E'$
    ConnectivityEstimation(H',$2k$)

We remark that recent work of Hariharan and Panigrahi [20] analyzes the same algorithm and shows that actually setting $\rho = O(\log(n)/\epsilon^2)$ is sufficient, whereas we set $\rho = O(\log^2(n)/\epsilon^2)$. Therefore the size of their sparsfier is only $O(n \log^2(n)/\epsilon^2)$.

H.2 Finding $O(n \log^2 n)$-size sparsifiers for graphs with polynomial weights

In this section, we describe another algorithm for computing κ_e which has the advantage that the computed κ_e's satisfy $\sum_{e \in E} u_e \kappa_e = O(n)$. By the last statement of Theorem 1.1, the size of the sparsifier would be $O(n \log^2 n)$. The algorithm, given in Algorithm 5, is a slight variation of the Estimation algorithm in [5].

The algorithm is based on finding k-partitions. A *k-partition* of a graph $G = (V, E)$ is a set $E' \subseteq E$ of edges that includes all $(k+1)$-light edges such that $|E'| \leq 2k(r-1)$ if $G - E'$ has r components. A k-partition is a "sparser" version of k-certificate as a k-certificate can have $k(n-1)$ edges.

Lemma H.1 (Benczúr and Karger [5]). *There is an algorithm Partition that outputs a k-partition in $O(m)$ time for unweighted graphs and $O(m \log n)$ time for graphs with arbitrary weights.*

We compute the κ_e's by using the ConnectivityEstimation procedure below. It is almost the same as the Estimation procedure in [5], which is for finding lower bounds of edge strengths. The only difference is that they call a WeakEdges procedure to find the k-weak edges instead of calling Partition to find k-light edges.

Lemma H.2. *After a call to ConnectivityEstimation(G,1), all the labels κ_e satisfy $\kappa_e \leq k_e$.*

Proof. The proof is the same as the proof of Corollary 4.8 in [5]. ■

Lemma H.3. *The values κ_e output by ConnectivityEstimation satisfy $\sum 1/\kappa_e = O(n)$.*

Proof. The proof is the same as the proof of Lemma 4.9 in [5]. ■

Lemma H.4. *ConnectivityEstimation runs in $O(m \log^2 n)$ time on a graph with polynomial weights.*

Proof. For a graph with polynomial weights, the maximum connectivity is bounded by some fixed polynomial, so there are at most $O(\log n)$ levels of recursion. The total number of edges in all the input graphs to Partition at each level of recursion is at most m. By Lemma H.1, each level of computation takes $O(m \log n)$ time. ■

Suppose we sample as described in Theorem 1.1. By Theorem 1.1 and Lemma H.3, the size

of the sparsifier is $O(n \log^2 n/\epsilon^2)$. The time needed for finding κ_e's is $O(m \log^2 n)$ and the time needed for finding the weight of every edge is $O(n \log^4 n/\epsilon^4)$. Therefore, the total running time is $O(m \log^2 n + n \log^4 n/\epsilon^4)$.

H.3 Finding $O(n \log n)$-size sparsifiers for graphs with polynomial weights

In [5], Benczúr and Karger presented an algorithm that finds a sparsifier of size $O(n \log n/\epsilon^2)$ for graphs with arbitrary weights in $O(m \log^3 n)$ time. This can be combined with the algorithms in the last two sections to prove Theorem 1.9.

Suppose we are given a graph G. First we apply Theorem 1.7 to find a sparsifier G' which approximately preserves all cuts of G to within a multiplicative error of $1 \pm \epsilon/4$. G' has size $O(n \log^3 n/\epsilon^2)$ and this takes $O(m + n \log^5 n/\epsilon^4)$ time. Then we apply Theorem 1.8 to find a sparsifier G'' which is a $1 \pm \epsilon/4$-approximation of G'. G'' has size $O(n \log^2 n/\epsilon^2)$ and this takes $O(n \log^3 n \cdot \log^2 n/\epsilon^2 + n \log^4 n/\epsilon^4) = O(n \log^5 n/\epsilon^4)$ time. Finally, we use Benczúr and Karger's algorithm to obtain a sparsifier G''' that is a $1 \pm \epsilon/4$-approximation of G''. G''' has size $O(n \log n/\epsilon^2)$ and this takes $O(n \log^2 n/\epsilon^2 \cdot \log^3 n) = O(n \log^5 n/\epsilon^2)$ time.

Note that G''' approximately preserves all cuts of G to within a multiplicative error of $1 \pm \epsilon$. The total running time is $O(m + n \log^5 n/\epsilon^4)$.